In [1]:

```
import pandas as pd

import matplotlib

import numpy as np
import scipy as sp


import IPython
from IPython import display


import sklearn
```

In [2]:

```
df = pd.read_csv("UCI_Credit_Card.csv")
```

In [3]:

```
df.shape
```

Out[3]:

```
(30000, 25)
```

In [4]:

```
df.head()
```

Out[4]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AI
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0	
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0	
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0	15
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0	20
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0	20

5 rows × 25 columns

In [5]:
```
df.describe()
```

Out[5]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.00000
mean	15000.500000	167484.322667	1.603733	1.853133	1.551867	35.485500	-0.016700	-0.133767	-0.16620
std	8660.398374	129747.661567	0.489129	0.790349	0.521970	9.217904	1.123802	1.197186	1.19686
min	1.000000	10000.000000	1.000000	0.000000	0.000000	21.000000	-2.000000	-2.000000	-2.00000
25%	7500.750000	50000.000000	1.000000	1.000000	1.000000	28.000000	-1.000000	-1.000000	-1.00000
50%	15000.500000	140000.000000	2.000000	2.000000	2.000000	34.000000	0.000000	0.000000	0.00000
75%	22500.250000	240000.000000	2.000000	2.000000	2.000000	41.000000	0.000000	0.000000	0.00000
max	30000.000000	1000000.000000	2.000000	6.000000	3.000000	79.000000	8.000000	8.000000	8.00000

8 rows × 25 columns

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   ID                          30000 non-null  int64  
 1   LIMIT_BAL                   30000 non-null  float64
 2   SEX                         30000 non-null  int64  
 3   EDUCATION                   30000 non-null  int64  
 4   MARRIAGE                    30000 non-null  int64  
 5   AGE                         30000 non-null  int64  
 6   PAY_0                       30000 non-null  int64  
 7   PAY_2                       30000 non-null  int64  
 8   PAY_3                       30000 non-null  int64  
 9   PAY_4                       30000 non-null  int64  
 10  PAY_5                       30000 non-null  int64  
 11  PAY_6                       30000 non-null  int64  
 12  BILL_AMT1                   30000 non-null  float64
 13  BILL_AMT2                   30000 non-null  float64
 14  BILL_AMT3                   30000 non-null  float64
 15  BILL_AMT4                   30000 non-null  float64
 16  BILL_AMT5                   30000 non-null  float64
 17  BILL_AMT6                   30000 non-null  float64
 18  PAY_AMT1                    30000 non-null  float64
 19  PAY_AMT2                    30000 non-null  float64
 20  PAY_AMT3                    30000 non-null  float64
 21  PAY_AMT4                    30000 non-null  float64
 22  PAY_AMT5                    30000 non-null  float64
 23  PAY_AMT6                    30000 non-null  float64
 24  default.payment.next.month  30000 non-null  int64  
dtypes: float64(13), int64(12)
memory usage: 5.7 MB
```

In [7]:
```
print(pd.isnull(df).sum())
```

```
ID                            0
LIMIT_BAL                     0
SEX                           0
EDUCATION                     0
MARRIAGE                      0
AGE                           0
PAY_0                         0
PAY_2                         0
PAY_3                         0
PAY_4                         0
PAY_5                         0
PAY_6                         0
BILL_AMT1                     0
BILL_AMT2                     0
BILL_AMT3                     0
BILL_AMT4                     0
BILL_AMT5                     0
BILL_AMT6                     0
PAY_AMT1                      0
PAY_AMT2                      0
PAY_AMT3                      0
PAY_AMT4                      0
PAY_AMT5                      0
PAY_AMT6                      0
default.payment.next.month    0
dtype: int64
```

In [8]:
```
df
```

Out[8]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	...	88004.0	31237.0	15980.0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	...	8979.0	5190.0	0.0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	...	20878.0	20582.0	19357.0
29998	29999	80000.0	1	3	1	41	1	-1	0	0	...	52774.0	11855.0	48944.0
29999	30000	50000.0	1	2	1	46	0	0	0	0	...	36535.0	32428.0	15313.0

30000 rows × 25 columns

In [9]:
```
df1 = df.copy()
```

In [10]:
```
df1
```

Out[10]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	...	88004.0	31237.0	15980.0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	...	8979.0	5190.0	0.0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	...	20878.0	20582.0	19357.0
29998	29999	80000.0	1	3	1	41	1	-1	0	0	...	52774.0	11855.0	48944.0
29999	30000	50000.0	1	2	1	46	0	0	0	0	...	36535.0	32428.0	15313.0

30000 rows × 25 columns

In [11]:
```
df1
```

Out[11]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	...	88004.0	31237.0	15980.0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	...	8979.0	5190.0	0.0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	...	20878.0	20582.0	19357.0
29998	29999	80000.0	1	3	1	41	1	-1	0	0	...	52774.0	11855.0	48944.0
29999	30000	50000.0	1	2	1	46	0	0	0	0	...	36535.0	32428.0	15313.0

30000 rows × 25 columns

In [12]:
```
df.drop('ID', axis=1, inplace=True)
```

In [13]:
```
df
```

Out[13]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	20000.0	2	2	1	24	2	2	-1	-1	-2	...	0.0	0.0	0.0
1	120000.0	2	2	2	26	-1	2	0	0	0	...	3272.0	3455.0	3261.0
2	90000.0	2	2	2	34	0	0	0	0	0	...	14331.0	14948.0	15549.0
3	50000.0	2	2	1	37	0	0	0	0	0	...	28314.0	28959.0	29547.0
4	50000.0	1	2	1	57	-1	0	-1	0	0	...	20940.0	19146.0	19131.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	220000.0	1	3	1	39	0	0	0	0	0	...	88004.0	31237.0	15980.0
29996	150000.0	1	3	2	43	-1	-1	-1	-1	0	...	8979.0	5190.0	0.0
29997	30000.0	1	2	2	37	4	3	2	-1	0	...	20878.0	20582.0	19357.0
29998	80000.0	1	3	1	41	1	-1	0	0	0	...	52774.0	11855.0	48944.0
29999	50000.0	1	2	1	46	0	0	0	0	0	...	36535.0	32428.0	15313.0

30000 rows × 24 columns

In [14]:
```
Target = df['default.payment.next.month']
```

In [15]:
```
Target.head()
```

Out[15]:
```
0    1
1    1
2    0
3    0
4    0
Name: default.payment.next.month, dtype: int64
```

In [16]:
```
df.drop('default.payment.next.month', axis=1, inplace=True)
```

In [17]:
```
df
```

Out[17]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT3	BILL_AMT4	BILL_AMT5
0	20000.0	2	2	1	24	2	2	-1	-1	-2	...	689.0	0.0	0.0
1	120000.0	2	2	2	26	-1	2	0	0	0	...	2682.0	3272.0	3455.0
2	90000.0	2	2	2	34	0	0	0	0	0	...	13559.0	14331.0	14948.0
3	50000.0	2	2	1	37	0	0	0	0	0	...	49291.0	28314.0	28959.0
4	50000.0	1	2	1	57	-1	0	-1	0	0	...	35835.0	20940.0	19146.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	220000.0	1	3	1	39	0	0	0	0	0	...	208365.0	88004.0	31237.0
29996	150000.0	1	3	2	43	-1	-1	-1	-1	0	...	3502.0	8979.0	5190.0
29997	30000.0	1	2	2	37	4	3	2	-1	0	...	2758.0	20878.0	20582.0
29998	80000.0	1	3	1	41	1	-1	0	0	0	...	76304.0	52774.0	11855.0
29999	50000.0	1	2	1	46	0	0	0	0	0	...	49764.0	36535.0	32428.0

30000 rows × 23 columns

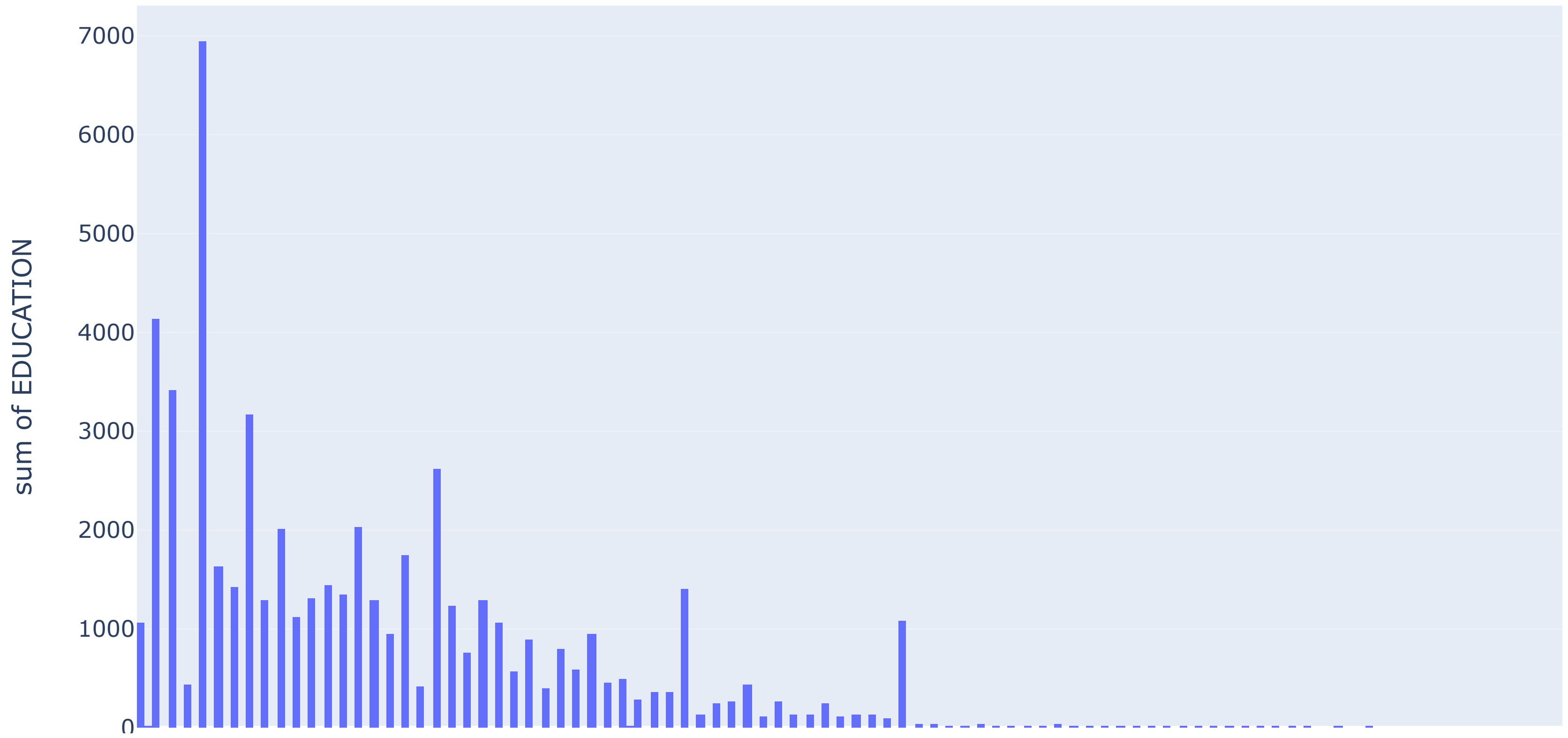
In [18]:
```
!pip install plotly matplotlib seaborn --quiet
```

In [19]:
```
import plotly.express as px
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (10, 6)
matplotlib.rcParams['figure.facecolor'] = '#00000000'
```

In [20]:
```
##Performing visualazation on the dataset

px.histogram(df, x='LIMIT_BAL', y = 'EDUCATION')
```



In [21]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [22]:
```
?MinMaxScaler
```

In [23]:
```
scaler = MinMaxScaler()
```

In [24]:
```
!pip install scikit-learn --upgrade --quiet
```

In [25]:
```
from sklearn.model_selection import train_test_split
```

In [26]:
```
scaler.fit(df)
```

Out[26]:
```
MinMaxScaler()
```

In [27]:
```
df1
```

Out[27]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	...	88004.0	31237.0	15980.0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	...	8979.0	5190.0	0.0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	...	20878.0	20582.0	19357.0
29998	29999	80000.0	1	3	1	41	1	-1	0	0	...	52774.0	11855.0	48944.0
29999	30000	50000.0	1	2	1	46	0	0	0	0	...	36535.0	32428.0	15313.0

30000 rows × 25 columns

In [28]:
```
train_inputs = scaler.transform(df)
```

In [29]:
```
import numpy as np
```

```
In [30]: print('train_inputs:', train_inputs.shape)

         train_inputs: (30000, 23)

In [31]: from sklearn.linear_model import LogisticRegression

In [32]: ?LogisticRegression

In [33]: model = LogisticRegression(solver='liblinear')

In [38]: model.fit(train_inputs, Target)

Out[38]: LogisticRegression(solver='liblinear')

In [39]: print(model.coef_.tolist())

         [[-0.9594440335818961, -0.10440421092724593, -0.6223602635972015, -0.4799834862912161, 0.429844836359249, 5.7
         37035715061759, 0.9187205049158633, 0.7917679618425738, 0.19935302096068125, 0.3456988373286697, 0.1058568026
         9962569, -2.0141366254711817, -0.17029937412430862, -0.4678703188893551, -0.21784092330992533, 0.368864151742
         837, 0.3238596503650357, -3.3021550485904796, -2.2274160898422823, -1.5791957306668691, -1.794122817258403, -
         1.4306618520190948, -1.223023993304811]]

In [40]: print(model.intercept_)

         [-2.01512022]

In [41]: train_preds = model.predict(train_inputs)

In [42]: train_preds

Out[42]: array([1, 0, 0, ..., 1, 0, 0], dtype=int64)

In [45]: train_targets = Target.copy()
         test_targets = Target.copy()
         val_targets =  Target.copy()
```

In [46]:
```
train_targets
```

Out[46]:
```
0        1
1        1
2        0
3        0
4        0
        ..
29995    0
29996    0
29997    1
29998    1
29999    1
Name: default.payment.next.month, Length: 30000, dtype: int64
```

In [47]:
```
train_probs = model.predict_proba(train_inputs)
train_probs
```

Out[47]:
```
array([[0.49724509, 0.50275491],
       [0.85169574, 0.14830426],
       [0.79488522, 0.20511478],
       ...,
       [0.17459052, 0.82540948],
       [0.75486279, 0.24513721],
       [0.73165155, 0.26834845]])
```

In [48]:
```
model.classes_
```

Out[48]:
```
array([0, 1], dtype=int64)
```

In [49]:
```
from sklearn.metrics import accuracy_score
```

In [50]:
```
accuracy_score(train_targets, train_preds)
```

Out[50]:
```
0.8102333333333334
```

In [51]:
```
from sklearn.metrics import confusion_matrix
```

In [52]:
```
confusion_matrix(train_targets, train_preds, normalize='true')
```

Out[52]:
```
array([[0.97440507, 0.02559493],
       [0.7677818 , 0.2322182 ]])
```

In [1]:

In [ ]: